Функции и параметры 2

Пример

Пример задачи

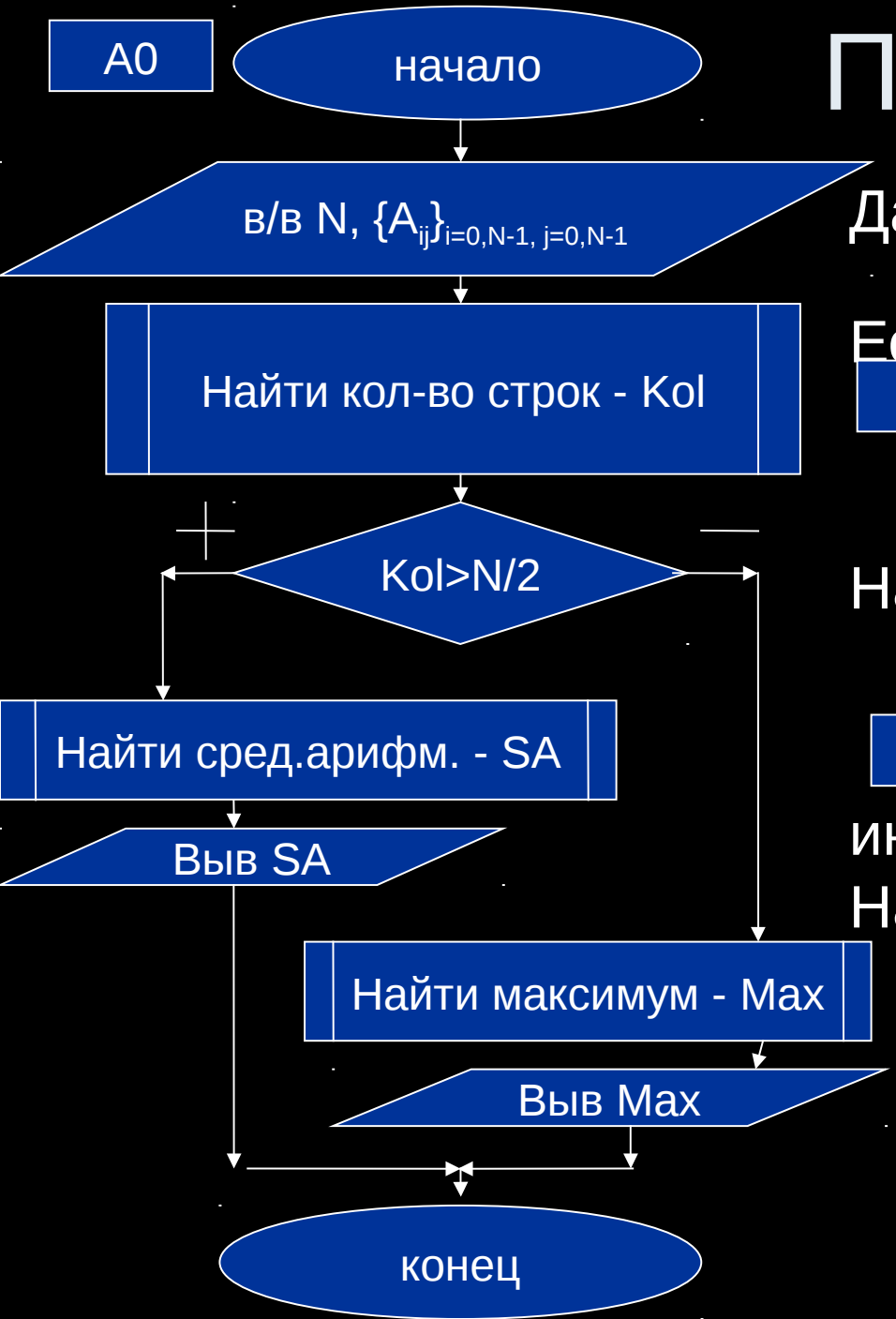
Дана квадратная матрица A размером N*N.

Если количество строк, в которых находятся только положительные элементы, больше N/2,

Найти среднее арифметическое значений всех положительных элементов,

иначе –

Найти максимальное значение среди элементов, лежащих выше главной диагонали матрицы **и на ней**.



Пример задачи

Дана квадратная матрица А размером N*N.

A0

Если количество строк, в которых находятся только положительные элементы больше N/2,

A0.1

A0.1 A0.2 A0.3

Найти среднее арифметическое значений всех положительных элементов,

A0.1.1 - Проверка строки

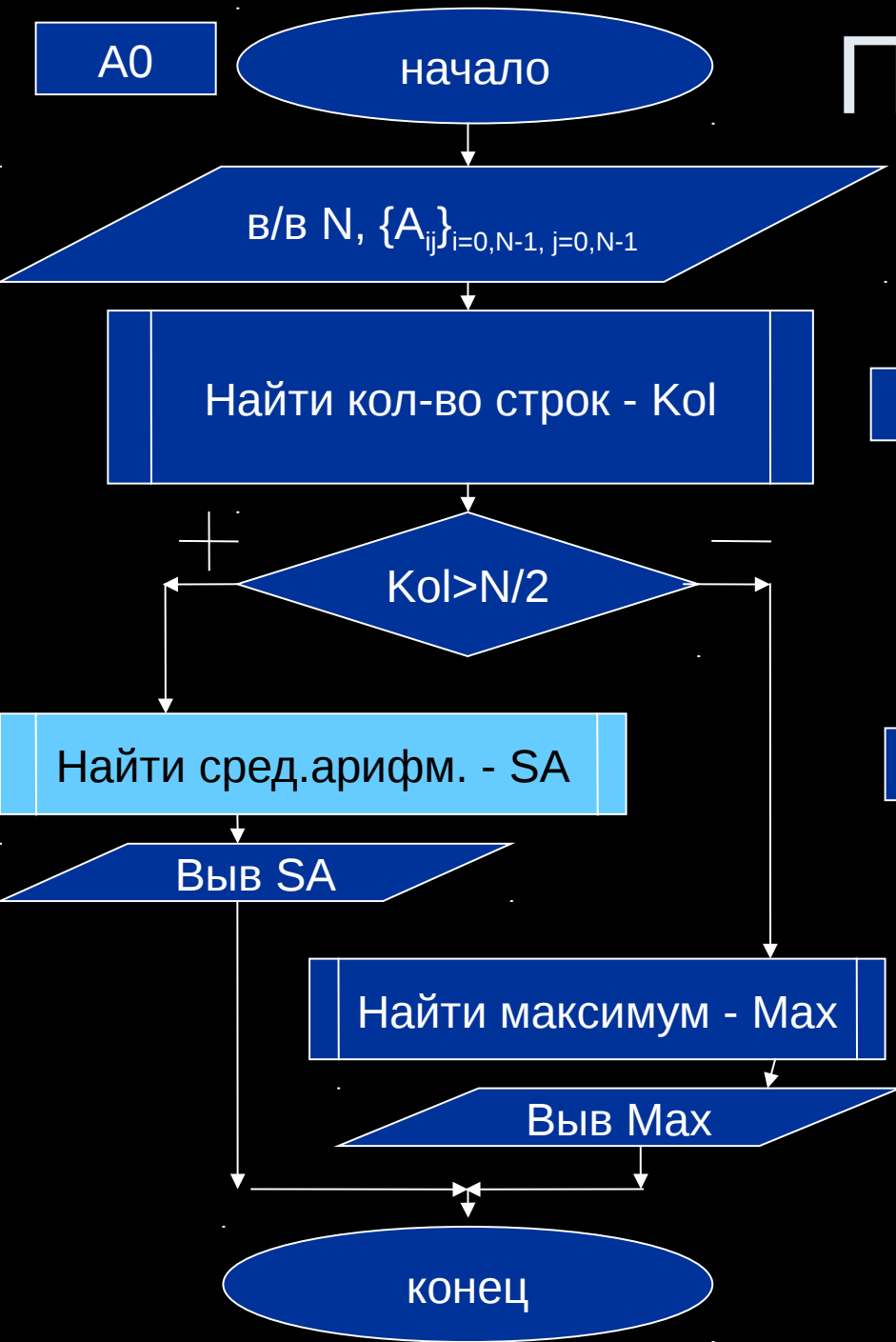
A0.2

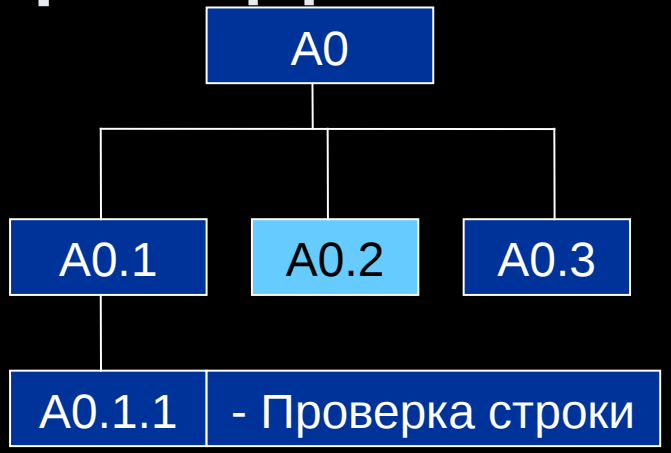
иначе –

Найти максимальное значение среди элементов, лежащих выше главной диагонали матрицы **и на ней**.

A0.3

Пример задачи





Дана квадратная матрица A размером N*N.

Если количество строк, в которых находятся только положительные элементы, больше N/2,

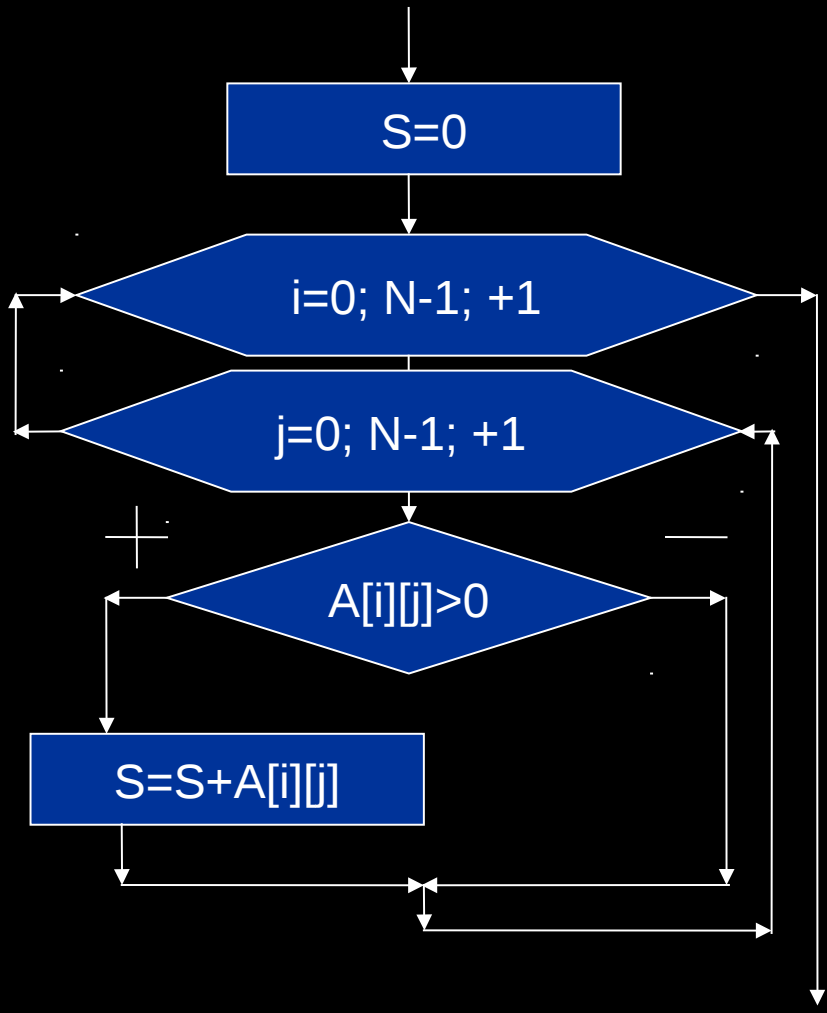
Найти среднее арифметическое значений всех положительных элементов,

иначе –

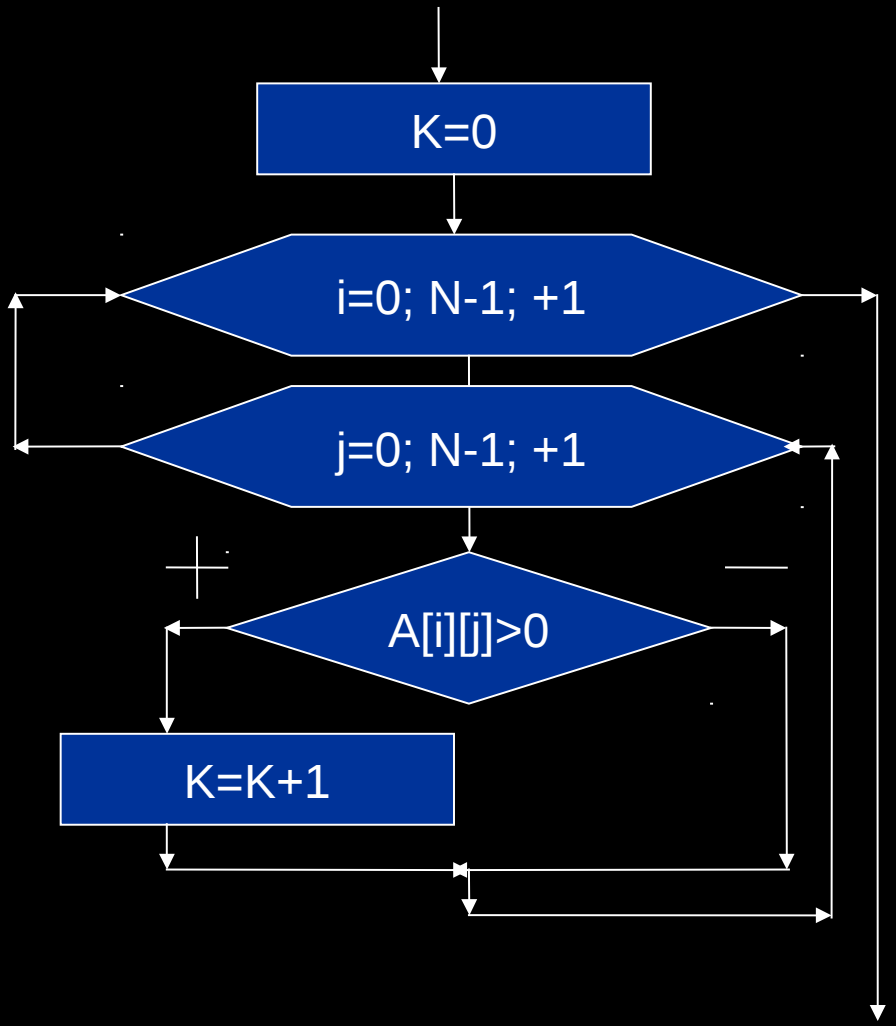
Найти максимальное значение среди элементов, лежащих выше главной диагонали матрицы **и на ней**.

A0.2 - Найти сред арифм Aij>0

Найти сумму - S



Найти количество - K



Т.о. для матрицы – удваиваем циклы

A0.2 - Найти сред арифм Aij>0



Найти сумму S и

Найти количество K одновременно

K>0, т.к. строк с положительными элементами > N/2

A0.2 - Найти сред.арифм Aij>0

ТИП FindSA(_____параметры_____){
_____локальные переменные_____

```
  S=0;    K=0;
  for (int i=0; i< N;i++)
     for (int j=0; j<N; j++)
        if (A[i][j]>0){
           S=S+A[i][j];
           K=K+1;
        }
  SA=S/K;
}
```

	вх	пром	вых	
S	-	+	-	лок
K	-	+	-	лок
i	-	+	-	лок
N	+	-	-	По знач.
j	-	+	-	лок
A	+	-	-	По знач.

Параметры и локал.переменные

вх	измен	вых		
-	+	-	лок	0) Локальная переменная
+	+	-	пар	1) По умолчанию, передача по значению (создается и изменяется копия переданного)
+	+	+	пар	2) Передача по адресу/ссылке Либо по значению + результат
+	-	-	пар	3) с const - Запрет на изменение, или как 1) (копия)
-	+	+	пар	4) Передача по адресу/ссылке Либо лок.перем.+ результат

A0.2 - Найти сред.арифм Aij>0

```
#define Nmax 10
void FindSA(const int N, int A[][Nmax], double *SA){
 int S,K; //локальные переменные

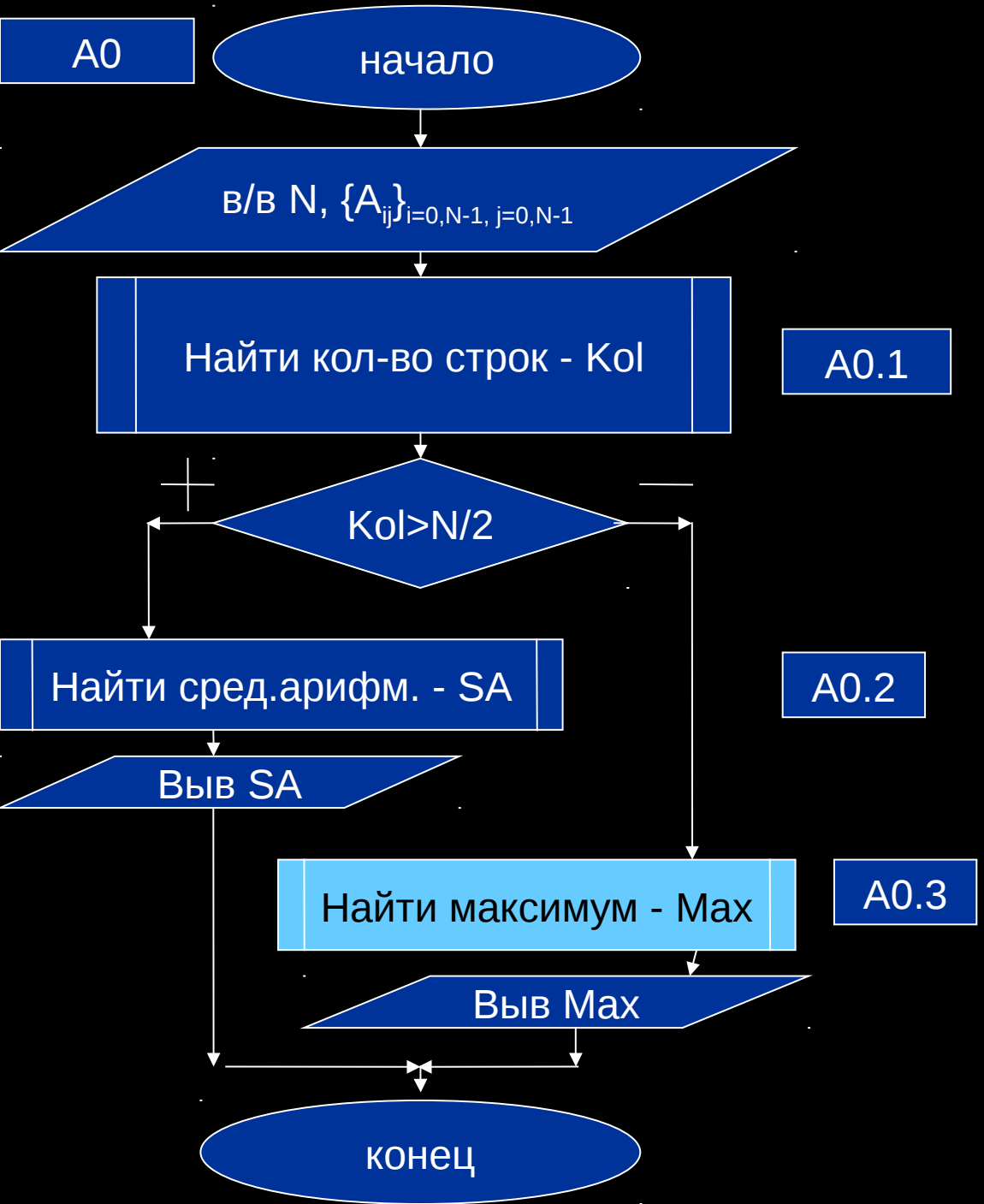
   S=0;    K=0;
   for (int i=0; i< N;i++)
      for (int j=0; j<N; j++)
         if (A[i][j]>0){
            S=S+A[i][j];
            K=K+1;
         }
   *SA=(double)S/K;
}
```

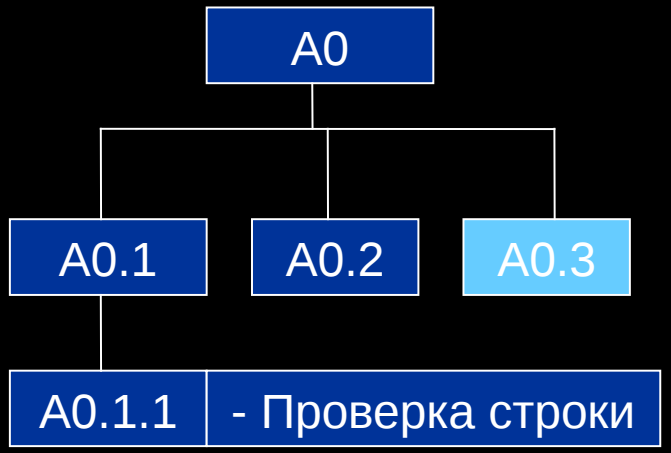
	вх	пром	вых	
S	-	+	-	лок
K	-	+	-	лок
i	-	+	-	лок
N	+	-	-	const/-
j	-	+	-	лок
A	+	-	-	const/-

A0.2 - Найти сред.арифм Aij>0

```
#define Nmax 10
double FindSA( const int N, int A[][Nmax]){
 int S,K; //лок. Переменные
 double SA;
   S=0;    K=0;
   for (int i=0; i< N;i++)
      for (int j=0; j<N; j++)
         if (A[i][j]>0){
            S=S+A[i][j];
            K=K+1;
         }
   SA=(double)S/K;
   return SA;
}
```

	вх	пром	вых	
S	-	+	-	лок
K	-	+	-	лок
i	-	+	-	лок
N	+	-	-	По знач.
j	-	+	-	лок





Дана квадратная матрица A размером N*N.

Если количество строк, в которых находятся только положительные элементы, больше N/2,

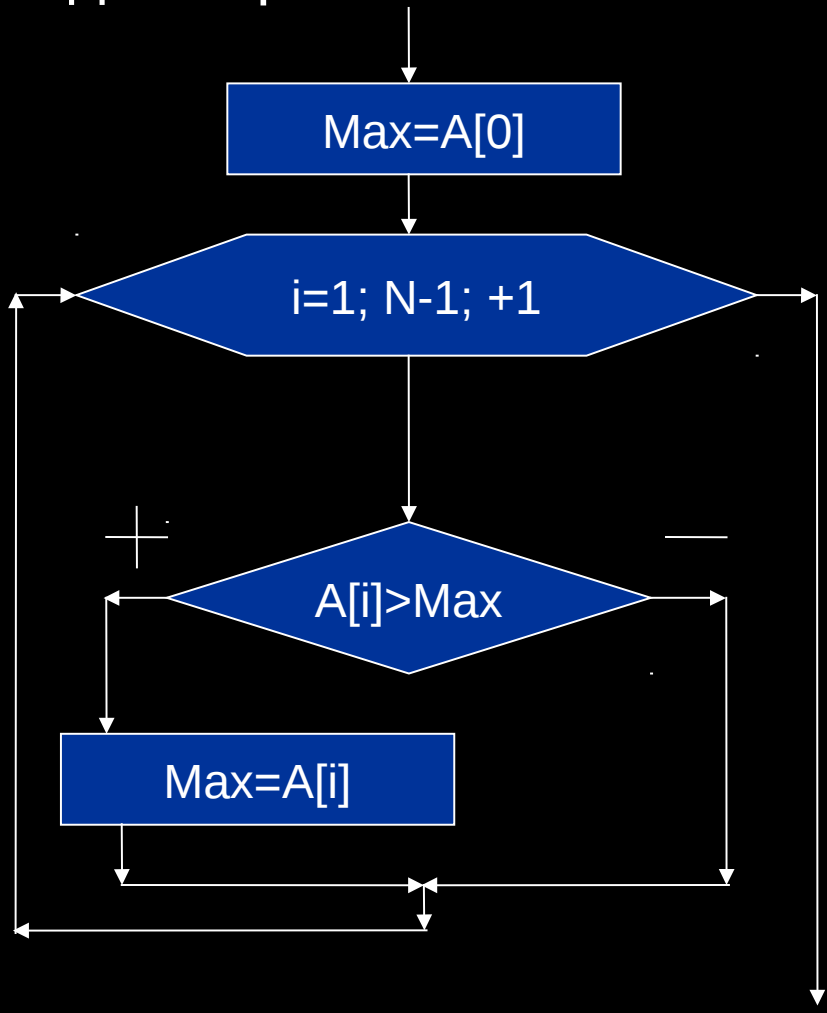
Найти среднее арифметическое значений всех положительных элементов,

иначе –

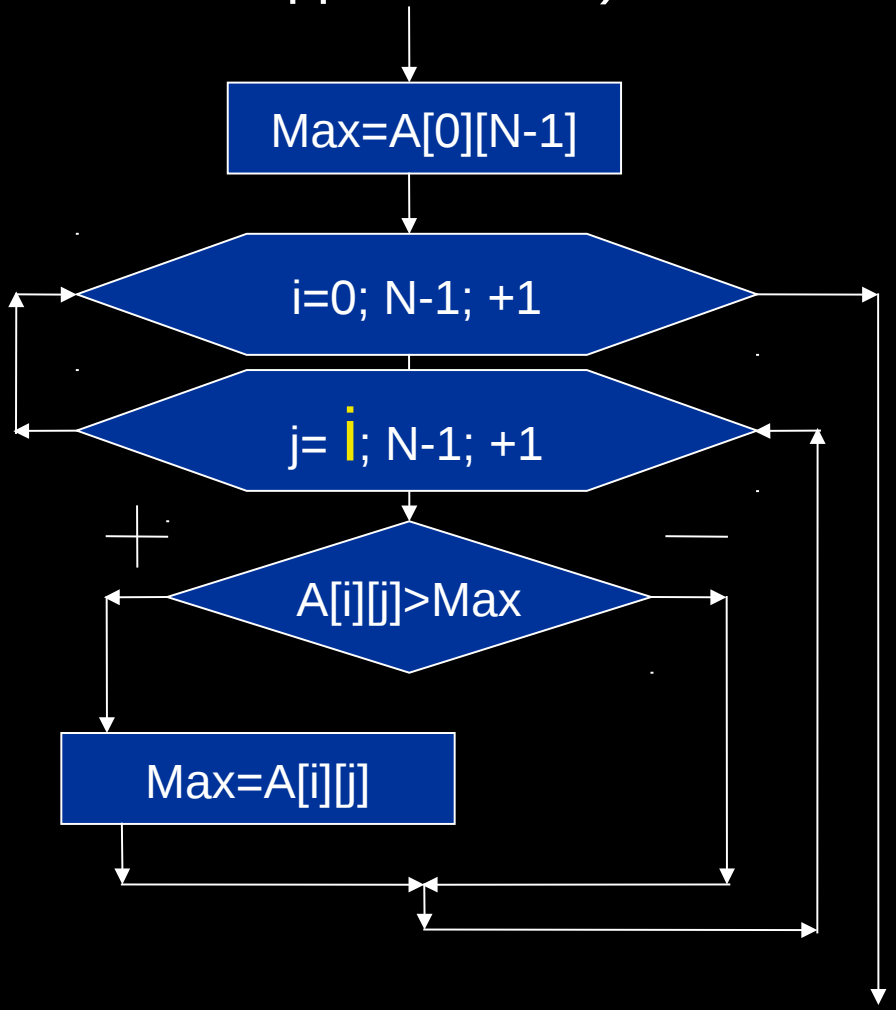
Найти максимальное значение среди элементов, лежащих выше главной диагонали матрицы и на ней.

A0.3 - Найти Max Aij | i<=j

Найти максимум в одномерном массиве:



Для матрицы (выше глав. диагонали):



Из spkmmu.doc

A0.3 - Найти Max Aij | i<=j

ТИП FindMax(_____параметры_____){
_____локальные переменные_____

```
  Max=A[0][N-1];
  for (int i=0;i<N;i++)
     for (int j=i; j<N;j++)
        if (A[i][j]>Max )
           Max=A[i][j];
}
```

	вх	измен	вых	
Max	-	+	+	По адресу/ссылке (C/C++) либо Результат функции
i	-	+	-	лок
N	+	-	-	const/-(копия)
j	-	+	-	лок
A	+	-	-	const/-(копия)

A0.3 - Найти Max Aij | i<=j

```
#define Nmax 10
void FindMax( const int N, const int A[][Nmax], int *Max){
  ///локальные переменные i,j

   *Max=A[0][N-1];
   for (int i=0;i<N;i++)
      for (int j=i; j<N;j++)
         if (A[i][j]>*Max )
            *Max=A[i][j];
}
```

	вх	измен	вых	
Max	-	+	+	По адресу/ссылке (C/C++) либо Результат функции
i	-	+	-	лок
N	+	-	-	const/-(копия)
j	-	+	-	лок
A	+	-	-	const/-(копия)

Max – указатель, хранит адрес памяти, где максимум

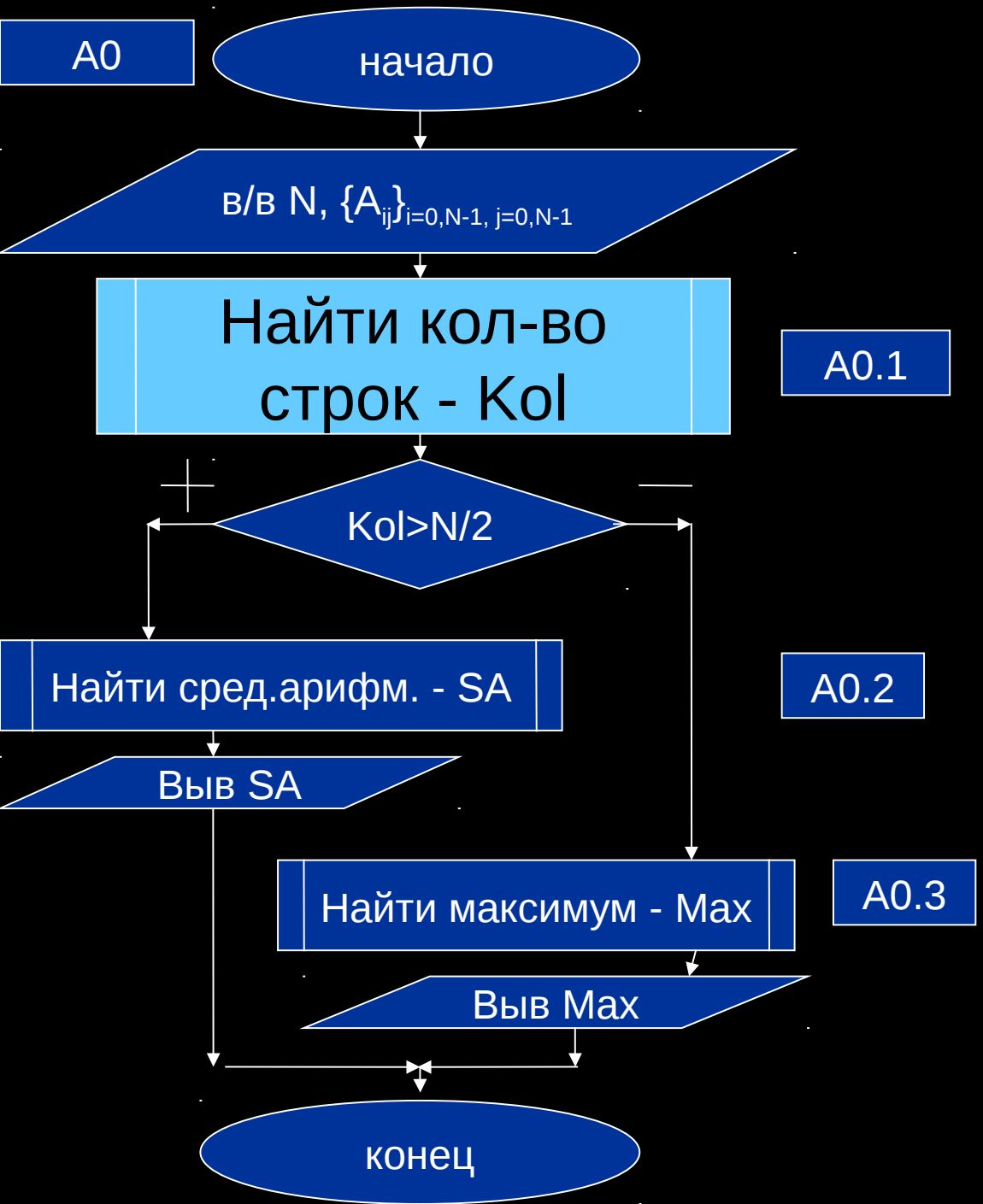
*Max – место/значение по адресу, где максимум

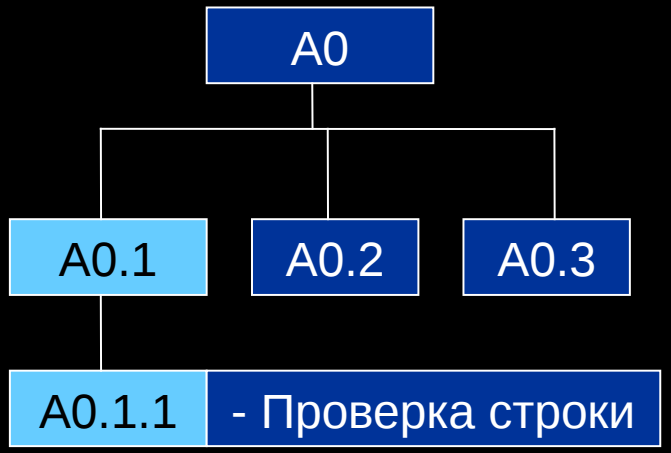
A0.3 - Найти Max Aij | i<=j

```
#define Nmax 10
int FindMax( const int N, const int A[][Nmax]){
  int Max; // локальные переменные i,j

   Max=A[0][N-1];
   for (int i=0;i<N;i++)
      for (int j=i; j<N;j++)
         if (A[i][j]>Max )
            Max=A[i][j];
    return Max;
}
```

	вх	измен	вых	
Max	-	+	+	По адресу/ссылке (C/C++) либо Лок.перем.+ рез-т функции
i	-	+	-	лок
N	+	-	-	const/-(копия)
j	-	+	-	лок
A	+	-	-	const/-(копия)





Дана квадратная матрица A размером N*N.

Если количество строк, в которых находятся только положительные элементы, больше N/2,

Найти среднее арифметическое значений всех положительных элементов,

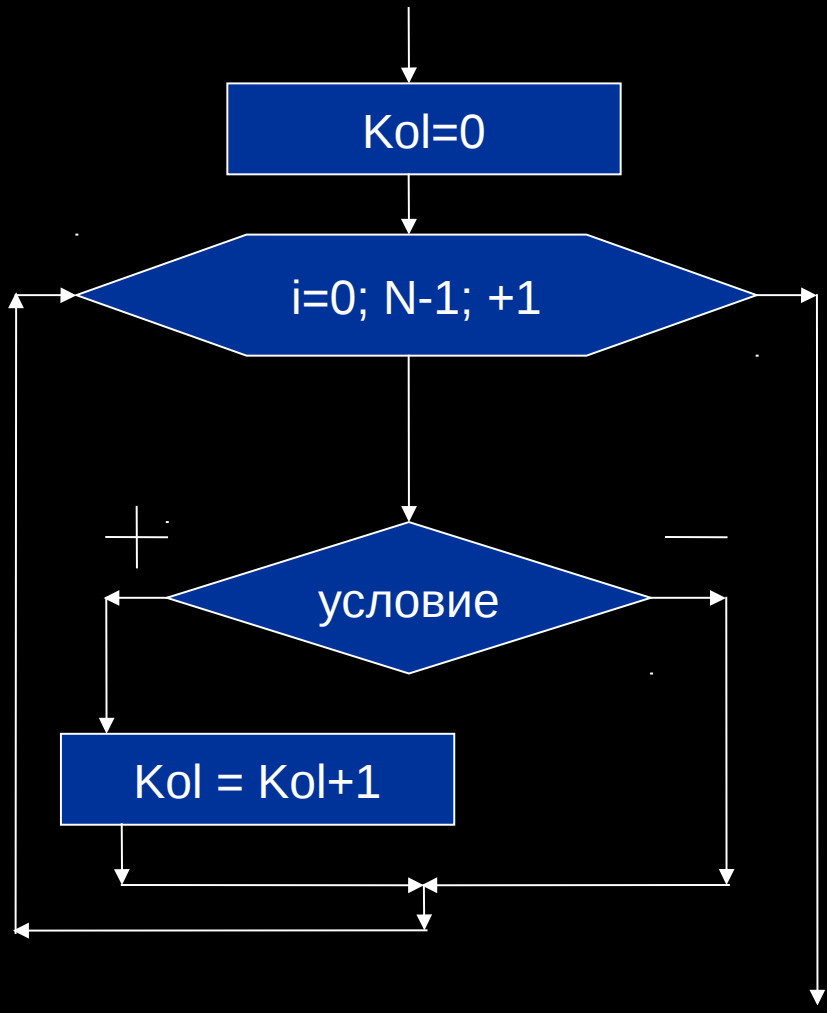
иначе –

Найти максимальное значение среди элементов, лежащих выше главной диагонали матрицы **и на ней**.

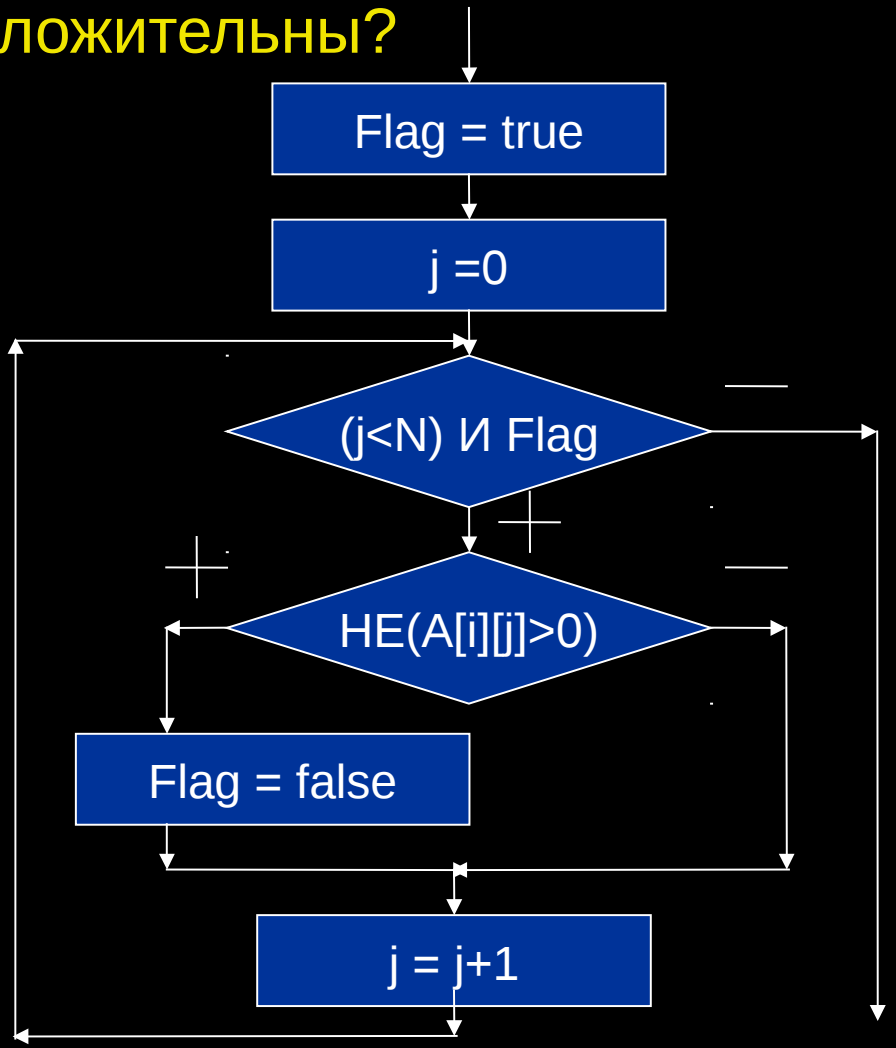
A0.1 - Найти количество строк,

в которых находятся только положительные элементы

Найти количество



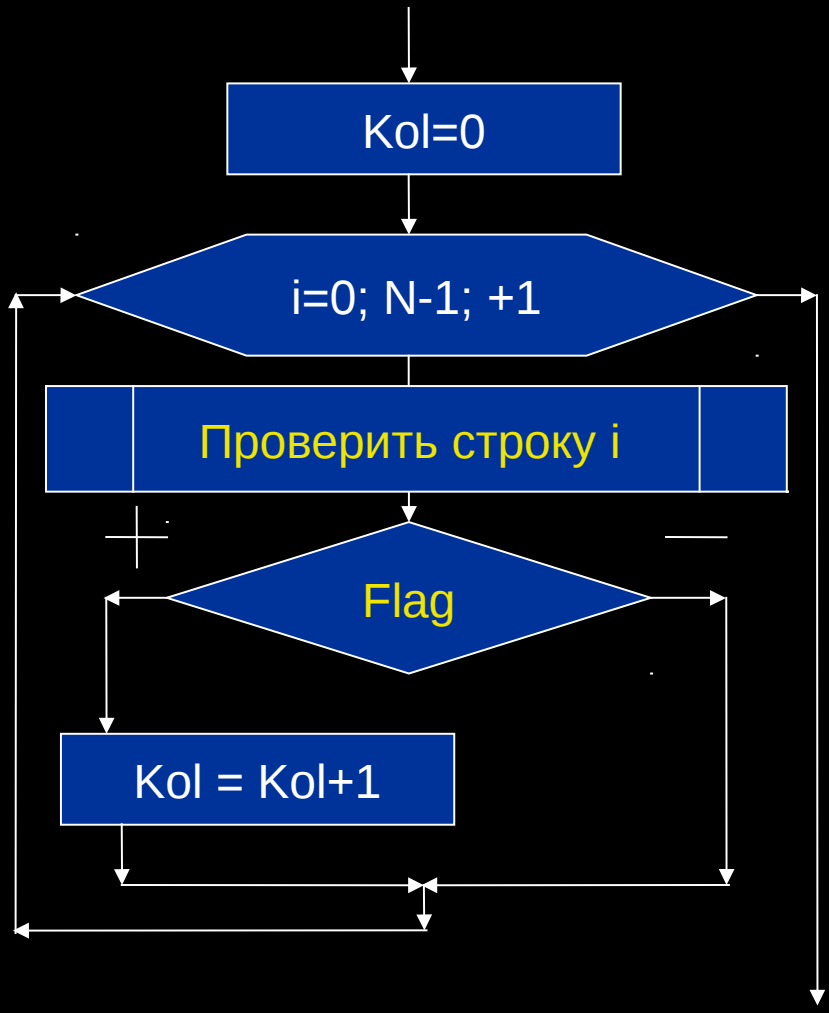
Проверить строку – ВСЕ ЛИ положительны?



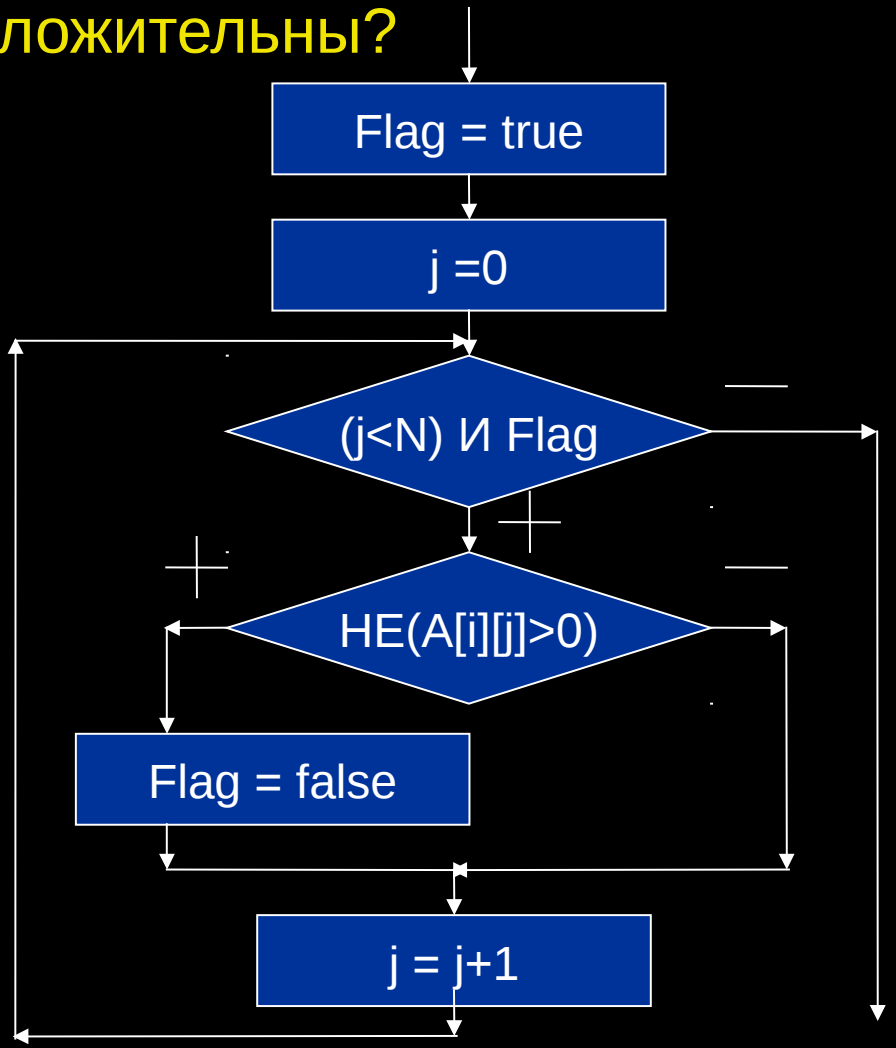
A0.1 - Найти количество строк,

в которых находятся только положительные элементы

Найти количество



Проверить строку – ВСЕ ЛИ положительны?



A0.1.1 – Проверить строку i

```
#define Nmax 10
void Check_i(const int N, const int i, int A[][Nmax],bool*Flag)
{
 int j;
   *Flag=true;    j=0;
   while (j<N && *Flag){
       if (! (A[i][j]>0) )
          *Flag= false;
      j++;
   }
}
```

	вх	измен	вых	
Flag	-	+	+	По адресу/ссылке (C/C++) либо Лок.перем.+ рез-т функции
i	+	-	-	const/-(копия)
N	+	-	-	const/-(копия)
j	-	+	-	лок
A	+	-	-	const/-(копия)

Flag – указатель, хранит адрес памяти, где флаг

*Flag – место/значение по адресу памяти, где флаг

A0.1.1 – Проверить строку i

```
#define Nmax 10
bool Check_i(const int N, const int i, int A[][Nmax]){
 int j;
 bool Flag = true;
   j=0;
   while (j<N && Flag){

      if (! (A[i][j]>0) )
         Flag= false;
      j++;
   }
   return Flag;
}
```

	вх	измен	вых	
Flag	-	+	+	По адресу/ссылке (C/C++) либо Лок.перем.+ рез-т функции
i	+	-	-	const/-(копия)
N	+	-	-	const/-(копия)
j	-	+	-	лок
A	+	-	-	const/-(копия)

A0.1 – Найти количество строк

```
#define Nmax 10
void FindKol(  const int N, int A[][Nmax], int *Kol){
   int i; bool Flag;

   *Kol=0;
   for (i=0; i< N;i++){
      Check_i( N,i, A, &Flag);
      if (Flag)
         (*Kol)++;
   }
}
```

	вх	пром	вых	
Kol	-	+	+	По адресу/ссылке (C/C++) либо Лок.перем.+ рез-т функции
Flag	-	+	-	лок
i	-	+	-	лок
N	+	-	-	const/-(копия)
A	+	-	-	const/-(копия)

&Flag – адрес переменной Flag передаётся

*Kol – место/значение по адресу памяти, где кол-во

Kol – указатель, хранит адрес памяти, где кол-во

A0.1 – Найти количество строк

```
#define Nmax 10
int FindKol(     const int N, int A[][Nmax]){
   int i; bool Flag;
   int Kol;
   Kol=0;
   for (i=0; i< N;i++){

      Flag=Check_i( N,i, A);
      if (Flag)
         Kol++;
   }
   return Kol;
}
```

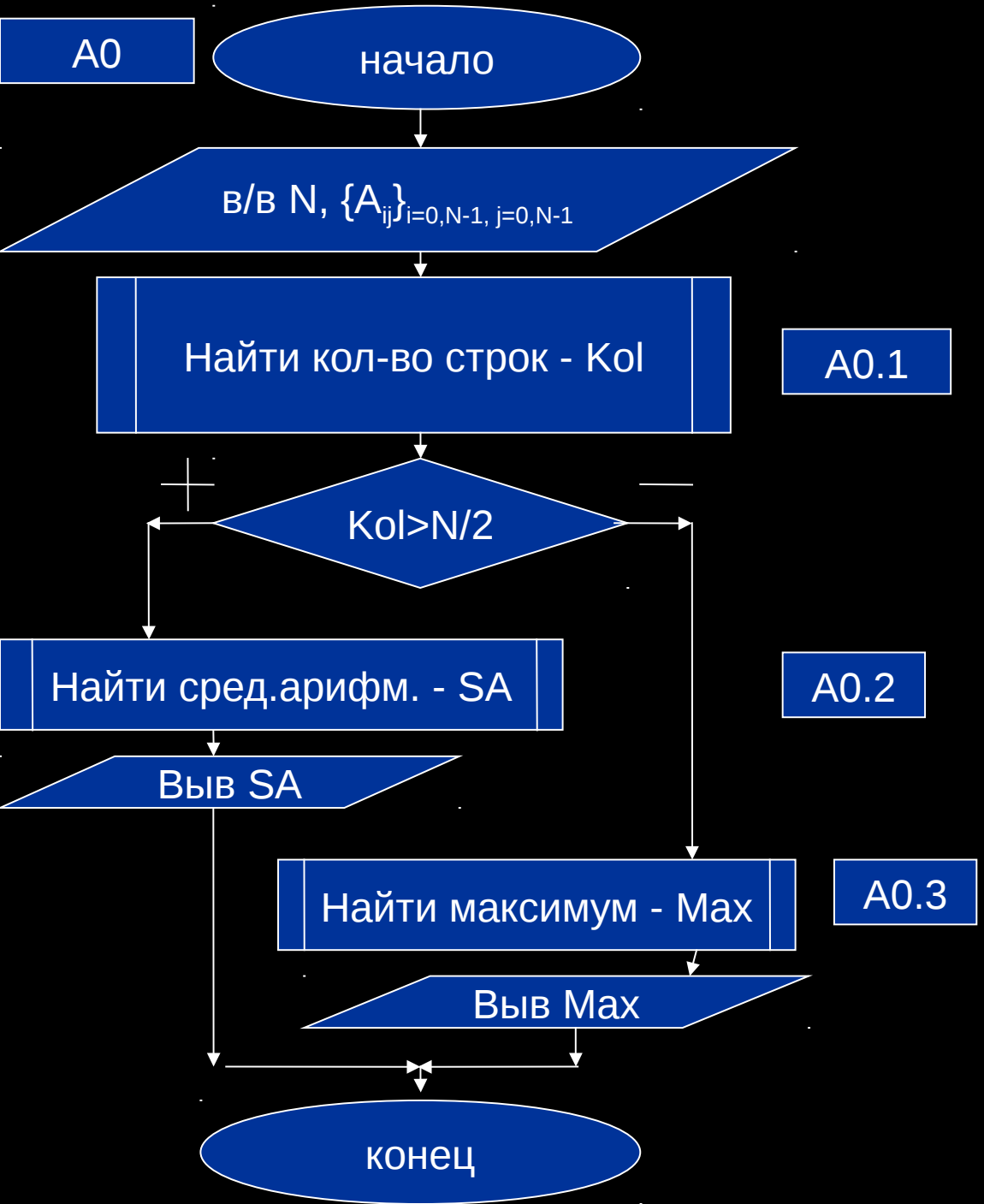
	вх	пром	вых	
Kol	-	+	+	По адресу/ссылке (C/C++) либо Лок.перем.+ рез-т функции
Flag	-	+	-	лок
i	-	+	-	лок
N	+	-	-	const/-(копия)
A	+	-	-	const/-(копия)

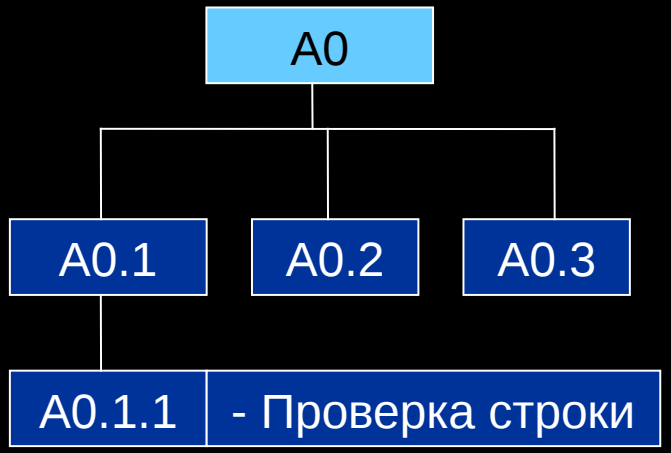
A0.1 – Найти количество строк

```
#define Nmax 10
int FindKol(     const int N, int A[][Nmax]){
   int i; bool Flag;
   int Kol;
   Kol=0;
   for (i=0; i< N;i++){


      if (Check_i( N,i, A))
         Kol++;
   }
    return Kol;
}
```

	вх	пром	вых	
Kol	-	+	+	По адресу/ссылке (C/C++) либо Лок.перем.+ рез-т функции
Flag	-	+	-	лок
i	-	+	-	лок
N	+	-	-	const/-(копия)
A	+	-	-	const/-(копия)





Дана квадратная матрица A размером N*N.

Если количество строк, в которых находятся только положительные элементы, больше N/2,

Найти среднее арифметическое значений всех положительных элементов,

иначе –

Найти максимальное значение среди элементов, лежащих выше главной диагонали матрицы **и на ней**.

A0 – Основная программа

```
#include <...>

#define Nmax 10

bool Check_i( const int N, const int i,  int A[][Nmax]){...}

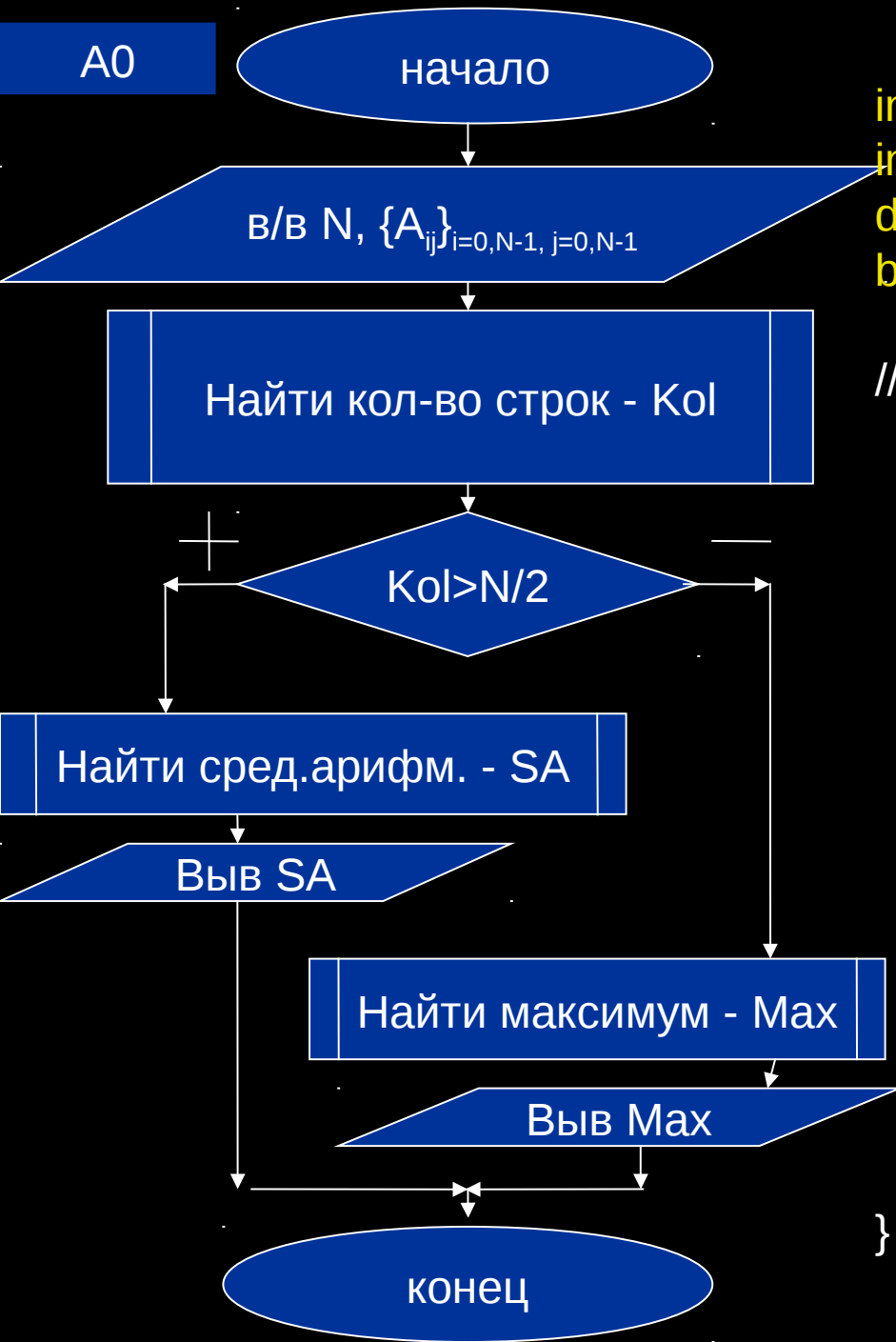
int FindKol( const int N, int A[][Nmax]){...};

int FindMax(const int N, int A[][Nmax]){...}

double FindSA(const int N, int A[][Nmax]){...}


int VvodVyvodNA( int *N, int A[][Nmax]){...}


int main(void){
int N, A[Nmax][Nmax], Kol, Max; double SA; bool Flag; …
}
```




```
int main(void){
int N, A[Nmax][Nmax], Kol, Max;
double SA;
bool Flag;

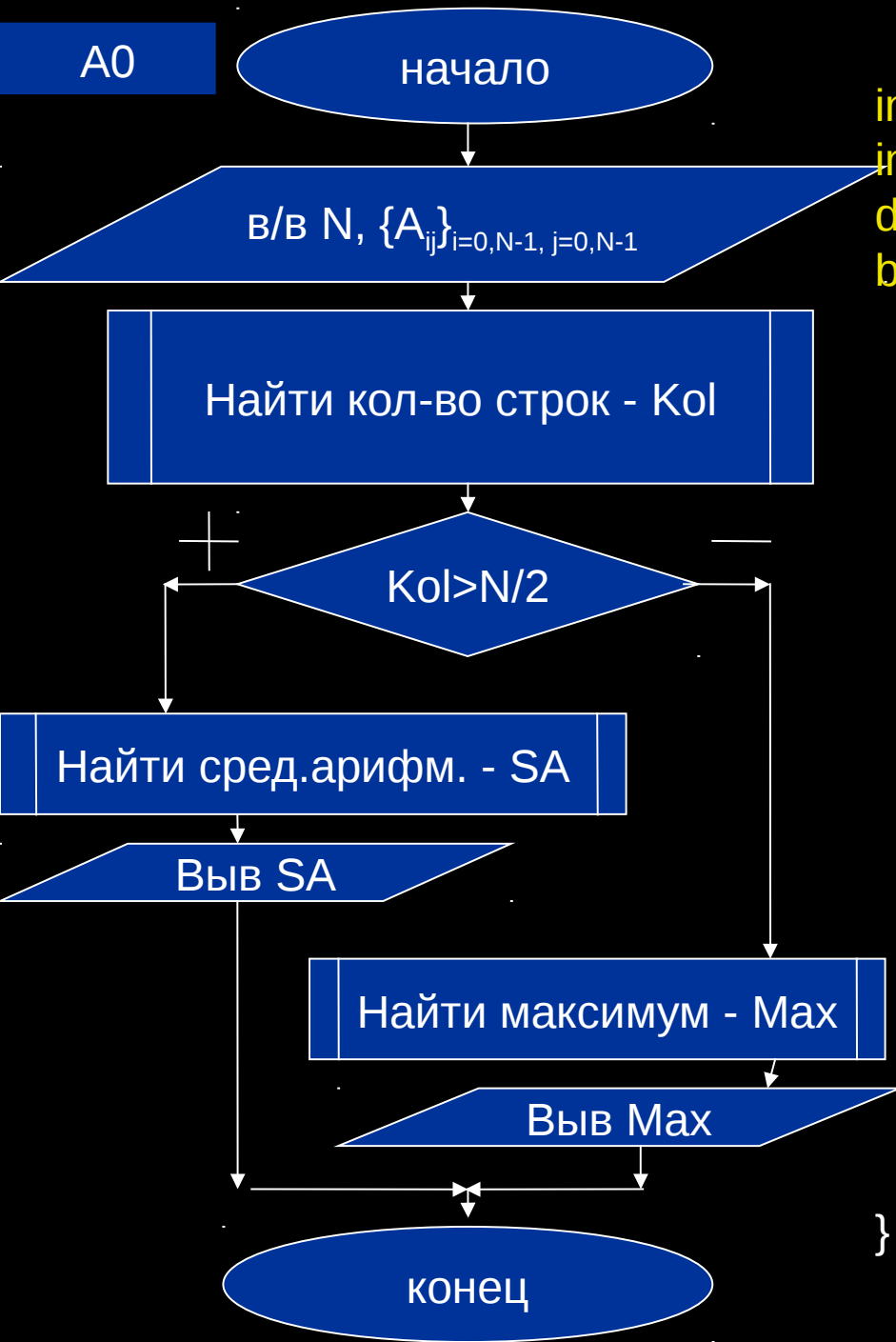
//-------------Б Ы Л О
   VvodVyvodNA( &N, A);
   FindKol( N, A, &Kol);
   if (Kol > N/2 ){

      FindSA( N, A, &SA);
      printf("Kol> N/2, Sa=%7.3lf", SA);
    }
   else{

      FindMax( N, A, &Max);
      printf("Kol<=N/2, Max=%d',Max);
    }

  return 0;

}
```




```
int main(void){
int N, A[Nmax][Nmax], Kol, Max;
double SA;
bool Flag;

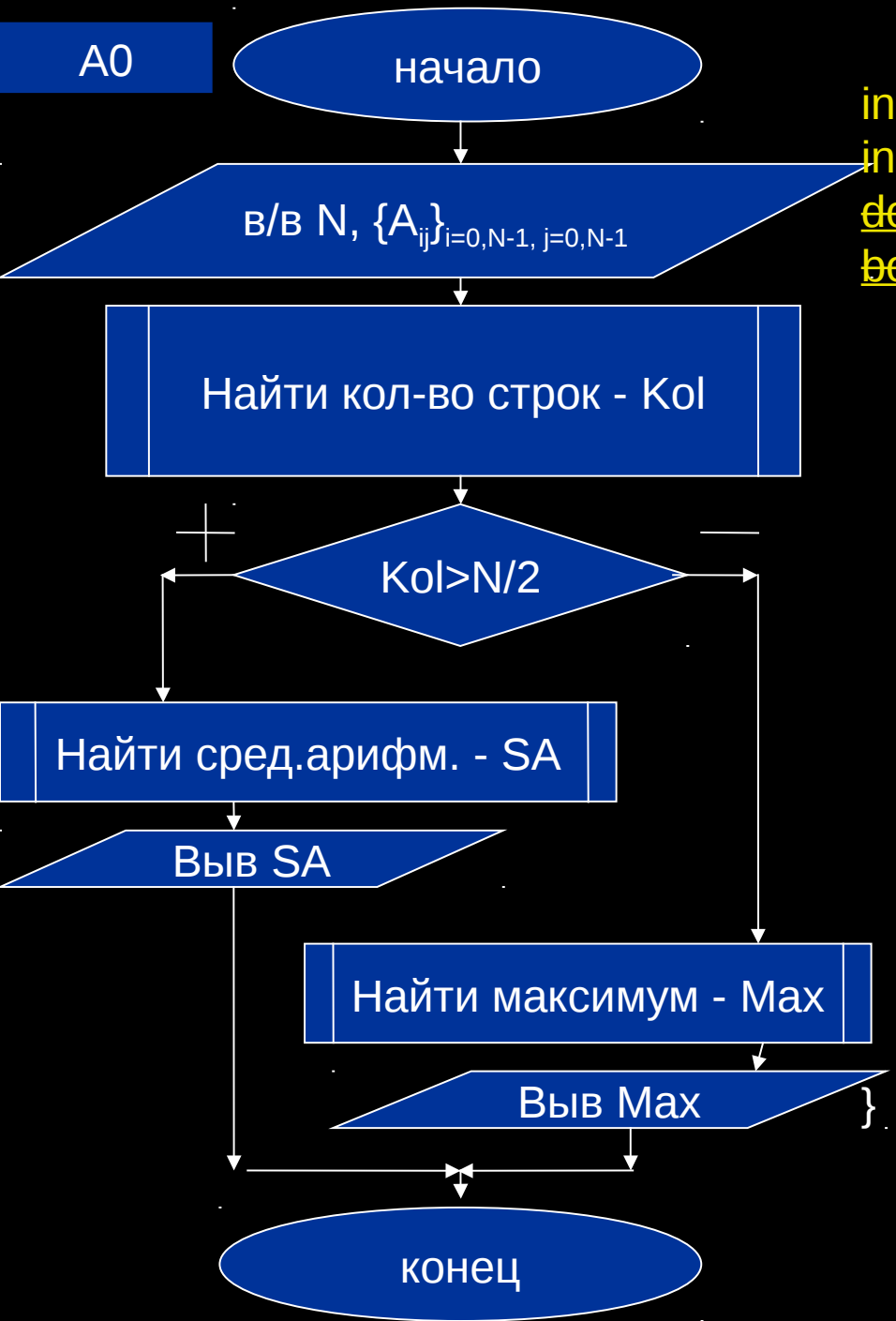

   N = VvodVyvodNA( A);
   Kol = FindKol( N, A);
   if (Kol > N/2 ){

      SA = FindSA( N, A);
      printf("Kol> N/2, Sa=%7.3lf", SA);
    }
   else{

      Max = FindMax( N, A);
      printf("Kol<=N/2, Max=%d',Max);
    }

  return 0;

}
```




```
int main(void){
int N, A[Nmax][Nmax], Kol, Max;
double SA;
bool Flag;

N = VvodVyvodNA( A);

if (FindKol( N, A) > N/2 ){

printf("Kol> N/2, Sa=%7.3lf", FindSA( N, A));
 }
else{

 printf("Kol<=N/2, Max=%d',FindMax( N, A));
 }

return 0;
}
```